8001

THE PATENTS ACT, 1970

IT IS HEREBY CERTIFIED THAT, the annex is a true copy of Application and Provisional specification filed on 26.7.2001 in respect of Patent Application No.721/Mum/2001 of Tata Consultancy Services(a Division of Tata Sons Limited) of Bombay House, Sir Homi Mody Street, Mumbai-400 023, Maharashtra, India , an Indian Company.

This certificate is issued under the powers vested on me under Section 147(1) of the Patents Act, 1970...........

.............Dated this 25th day of January 2002

[illegible]
(N.K.Garg)
Asst Controller of Patents & Designs

CERTIFIED COPY OF PRIORITY DOCUMENT

[redacted] 02-26-02 2123

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
Art Unit 2123 Examiner Unknown

[illegible]

In Re:	Sreedhar Sannareddy Reddy et al.
Case:	P8001
Serial No.:	10/052,071
Filed:	01/16/2002
Subject:	System and Apparatus for Programming System Views in an Object Oriented Environment

To: The Commissioner of Patents and Trademarks
Washington, D.C. 20231

Dear Sir,

Letter to the Examiner

Enclosed herewith is a certified copy of the provisional application from the Indian Patent Office which is a priority claim for the above-mentioned matter. Also enclosed is a copy of the Information Disclosure Statement that accompanied the US utility patent application submitted to the USPTO on January 16, 2002. Applicant desires that the certified copy of the provisional application be matched with the above referenced case.

Respectfully submitted,
Sreedhar Sannareddy Reddy et al.

[illegible]
Donald R. Boys
Reg. No. 35,074

Donald R. Boys
Central Coast Patent Agency, Inc.
P.O. Box 187
Aromas, CA 95004
(831) 726-1457

26 2001
मुंबई - ४०० ०१३.

FORM-1

THE PATENTS ACT, 1970

(39 OF 1970)

APPLICATION FOR GRANT OF A PATENT

(See sections 5(2), 7, 54 and 135 and rule 33A)

Electronic

Comp/P-1 Asstt. Controller
Pl. acceptance
[illegible]
26-07-01

1. ~~I~~/We, TATA CONSULTANCY SERVICES (a Division of TATA SONS LIMITED), of Bombay House, Sir Homi Mody Street, Mumbai 400 023, Maharashtra, India, an Indian Company

27 JUL 2001
7471
मुंबई-४०० ०१३.

2. hereby declare :-

(a) that ~~I am~~/we are in possession is an invention titled METHOD AND APPARATUS FOR DEVELOPMENT AND MAINTENANCE OF SOFTWARE SYSTEMS

(b) that the Provisional/~~Complete~~ specification relating to this invention is filed with this application

(c) that there is no lawful ground of objection to the grant of a patent to ~~me~~/us.

3. further declare that the inventor(s) for the said invention ~~is~~/are :

(a) SREEDHAR SANNAREDDY REDDY of Tata Consultancy Services, Hadapsar Industrial Estate, Pune 411 013, Maharashtra, India, an Indian National; and

(b) SAJEEV THOMAS of Tata Consultancy Services, Hadapsar Industrial Estate, Pune 411 013, Maharashtra, India, an Indian National

ORIGINAL

721/mum/2001

26/7/2001

721 | मुंबई MUM | 2001

26 JUL 2001

Received Rs. 5000/- in Cash
Cheque/M.O. on 26/7/01
Vide Entry 3/20 in
Register of [illegible]
[illegible]
26.07.01 [illegible]
Cashier

4. I/We, claim the priority from the application(s) filed in convention countries, particulars of which are as follows :

(a) [Country]

(b) [Appln.No.]

(c) [Date of Appln.]

(d) [Applicant in Convention Country] N.A.

(e) [Title]

5. I/We state that the said invention is an improvement in or modification of the invention, the particulars of which are as follows and of which I/We are the applicant/patentee :

(a) Application No.

(b) Date of application

N.A.

6. I/We state that the application is divided out of my/our application, the particulars of which are given below and pray that this application deemed to have been filed on date under section 16 of the Act.

(a) Application No.

(b) Date of filing of provisional specification : N.A.

and date of filing of complete specification :

7. That ~~I~~/We are the assignee or legal representative of the true and first inventors.

8. That ~~my~~/our address for service in India is as follows : **R.K. DEWAN & COMPANY,** Trade Marks & Patents Attorneys, 78, Podar Chambers, S.A.Brelvi Road, Fort, Mumbai 400 001, Maharashtra, India, Tel. (91) 22-2661662/2663002, Telefascimile number (91) 22-2650159, Email>rkdewan@vsnl.com.

भारत सरकार पेटेन्ट कार्यालय शाखा
26 JUN 2001
मुंबई - ४०० ०१३.

9. Following declaration was given by the inventor(s) ~~or applicant(s) in the convention country~~ :

~~I~~/We the true and first inventors for this invention ~~or the applicant(s) in the convention country~~ declare that the applicant(s) herein ~~i~~s/are ~~my~~/our assignee or legal representative :

[illegible]

(SREEDHARSANNAREDDY REDDY)

[illegible]

(SAJEEV THOMAS)

10. That to the best of ~~my~~/our knowledge, information and belief the fact and matters stated herein are correct and that there is no lawful ground of objection to the grant of patent to ~~me~~/us on this application

भारत सरकार पेटेन्ट कार्यालय शाखा
2 5 [illegible] 2001
मुंबई - ४०० ०१३.

11. Following are the attachment with the application :

(a) ~~Provisional~~/Complete specification (3 copies)

(b) Drawings (3 Copies)

(c) ~~Priority document(s)~~

(d) ~~Statement of Undertaking~~ on FORM-3

(e) Copy of General Power of authority

(f)

(g)

(h)

(i) Fee Rs. 5000/- In cash/cheque/bank draft bearing No.

date on Bank

~~I~~/We request that a patent may be granted to ~~me~~/us for the said invention.

Dated this 24th day of July 2001.

Signature : [illegible]

Name :(HEMANT DHANSUKHLAL VAKIL)

To :

The Controller of Patents

The Patent Office

at MUMBAI

भारत सरकार पेटेन्ट कार्यालय शाखा

मुंबई - ४०० ०१३.

FORM-2

THE PATENT ACT, 1970

PROVISIONAL

Specification

SECTION – 10

METHOD AND APPARATUS FOR DEVELOPMENT AND MAINTENANCE OF SOFTWARE SYSTEMS

TATA CONSULTANCY SERVICES,

(a Division of TATA SONS LIMITED),

of Bombay House, Sir Homi Mody Street,

Mumbai 400 023, Maharashtra, India,

an Indian Company

ORIGINAL

THE FOLLOWING SPECIFICATION DESCRIBES

THE NATURE OF THIS INVENTION :-

This invention relates a method and apparatus for development and maintenance of software systems.

Particularly, this invention relates to model repositories, which play a central role in the development and maintenance of large enterprise software systems.

A complex software system has many facets, many levels of abstraction and many views, like analysis view, design view, GUI view, database view, architectural view, and the like. A model repository is required to provide modeling abstractions to capture all these views in a consistent manner.

There are many commercial repository systems that provide extensibility around standard methodologies like Unified Modeling Language (UML), Entity Relationship (ER) modeling, etc. They typically also package visual modeling tools that support these standard methodologies either directly or in the form of a bridge to third party modeling tools (e.g.: Rational Rose).

An important aspect of modeling is the ability to view and edit models using a diagrammatic notation.

However, they do not provide any built-in support that enables end-users to easily specify visual diagrammatic notation for modeling abstractions introduced by them. They either have to be content with a generic, non-diagrammatic interface or build a graphical system from scratch if they want to provide visual modeling capability to abstractions introduced by them. This has long been an area where generic repository systems have been weak. Though they provide strong model extensibility features, the prior art apparatus and method do not provide any built-in support for rendering these extended models in a visual, diagrammatic language. This is seen as one of the handicaps affecting their wider acceptance.

In the prior art known to the invention, there is no standard modeling methodology that addresses all these aspects.

To overcome this in the prior art, an extensible meta modeling framework is provided in model repositories that enables users to extend the modeling system with abstractions of their own.

A principal object of this invention is to provide a method and apparatus that enables meta-model driven repository systems to provide a programmable visual modeling framework that enables end-users to specify a concrete graphical notation for abstract models, without having to program them from scratch.

The invention will now be described with reference to the accompanying drawings in which

Figure 1 illustrates a modeling framework in accordance with this invention;
Figure 2 illustrates an example of a meta model which forms the bases for creating models in accordance with the framework of Figure 2;
Figure 3 is a block diagram of a diagram model in accordance with this invention;

Figure 4 is a table showing the objects, association and properties contained in the meta model of figure 2;

Figure 5 Symbol definition in accordance with this invention;

Figure 6 Annotation definition in accordance with this invention;

Figure 7 ConnectorEnd Definition in accordance with this invention;

Figure 8 Connector Definition in accordance with this invention;

Figure 9 A template for mapping a symbol icon to an object in the model in accordance with this invention;

Figure 10 A template for mapping a container with containees : in accordance with this invention;

Figure 11 A template for defining an association connector in accordance with this invention;

Figure 12 A template for defining an object connector in accordance with this invention:

Figure 13 A template for defining a junction connector in accordance with this invention; and

Figure 14 shows a template for defining Nary connector in accordance with this invention.

Referring to the drawings the modeling framework (mappable to MOF) for generic, extensible repository systems, in accordance with this invention is shown in Figure 1 of the drawings and is self explanatory.

Figure 2. of the accompanying drawings shows a Meta meta model. The meta meta model (figure 2) is the base model. It is the schema for modeling meta models. The meta meta model is capable of describing itself, i.e., it can model itself. It is the root of the instantiation hierarchy. The meta meta model consists of objects, associations and properties shown in the table of Figure 4.

A meta model is an instance of the meta meta model. It consists of Meta Objects with associated Meta Properties, and Meta Associations defined between Meta Objects. A meta model defines the structure and semantics for the information system model (e.g.: UML meta model).

An information system model, also referred to as the user model, is an instance of a meta model. It captures the description of the information system from various points of view as specified by the meta model (e.g.: UML model of a banking system).

The above modeling framework is abstract enough to be able to support modeling techniques like UML, ER, etc.

A generic programmable diagramming framework, using the above modeling framework as the reference is hereinafter described.

In the first instance is described a model for diagramming, then a language for specifying visual attributes of diagram elements, a language for mapping diagram elements to meta-model elements, and finally a generic diagram drawing tool that interprets these specifications.

A diagram model created in accordance with the teachings of this specification is shown in Figure 3 of the drawings.

A diagram can be conceptualized as a set of connected symbols. A symbol can have text fields; a symbol can contain other symbols; a connector connects two symbols; a connector can have annotations. There is considerable similarity in structure between the meta-meta model (figure 2) and the diagram model (figure 3). A meta-object is typically shown as a symbol in the diagram, with associated meta-properties shown as fields in the symbol. A meta-association is typically shown as a connector, with either static annotations or annotations derived from associated meta-object' properties. A composite symbol (a container) can contain other symbols.

Diagrams are specified in terms of diagram types (e.g.: Class diagram, usecase diagram, etc). A diagram type is specified in terms of two files: a diagram specification file and a map specification file. In this section, we describe the diagram specification file, which specifies visual attributes of various symbols and icons that comprise the diagram type.

The diagram specification file defines symbols, connectors, connector ends and annotations that make up a diagram type. Symbols and connectors are collectively referred to as icons. Each icon is uniquely identified by an id called *IconId*.

Figure 5 is an example of symbol definition in accordance with this invention.

In property section seen in Figure 5, one can specify properties like icon bitmap to be displayed in the icon palette (described later), background color, etc.

In shape section, one can specify the drawing commands to draw the shape of the symbol. The following are the various drawing commands:

For each of the commands various options like line style, color, width, fill pattern, etc can be specified.

Boundary section specifies the boundary of the symbol. It consists of a polygon whose vertices are specified. The polygon is to enclose the shape of the symbol. Connector head and tail are drawn upto the polygon edges.

The Annotation definition is shown in Figure 6. This drawing describes the syntax to be used to define an annotation which will be used as part of the connector definition.

Figure 7 shows ConnectorEnd Definition in accordance with this invention.

In the property section, one can specify properties like background color, etc.In shape section, one can specify the drawing commands (listed above) to draw the shape of the annotation.

Figure 7 describes the syntax to be used to define a connector head or tail (referred to as connector end) which will be used as part of the connector definition. The syntax is identical for both. The head and tail will be drawn inside a rectangle whose dimensions have to be specified. Drawing commands are the same as the ones used in symbol definition.

Figure 8 shows Connector Definition in accordance with this invention This drawing describes the syntax to be used to define a connector.

In property section, one can specify properties like icon bitmap to be displayed in the icon palette (described later), connector ends to be used at the head/tail/middle of the connector, line style, color, width, and if the line style is double line then fill color, fill pattern, etc.

Figure 9 shows a template for mapping a symbol icon to an object in the model in accordance with this invention. In Figure 9 is described the map specification file that specifies the mapping between diagram elements of a diagram type and meta-model elements.

The map specification consists of the following sections. In the specification, *IconId* refers to the *IconId* specified in the diagram definition file. The specification has a provision to specify external scripts (like extern 'C') using *'script'* directive. This is an escape route to provide more complex behavior where required.

The symbol section associates a drawing sheet symbol with a repository meta-object.

objectprop section is for setting certain attributes automatically in the repository when an object of this type is created in the drawing sheet(by virtue of the object being of this type certain attributes have to be set - which are obvious to the user just by the look of the symbol). In addition, while opening an existing object it serves as a filter to select only those objects which match the values specified in the *objectprop* section.

The *symbolprop* section is for specifying which properties / associated objects of the repository object are to be displayed along with the icon in the diagram.

By using list-control, the associated object's information can be shown in a list box having multiple columns.

Figure 10 shows a template for mapping a container with containees : in accordance with this invention.

In Figure 10 the symbol can contain other symbols is specified. It also specifies the meta association between the meta-object mapped to the container symbol and the meta-object mapped to the containee symbol.

Figure 11 shows a template for defining an association connector in accordance with this invention: Figure 11 shows a method by which this invnetion maps a connector (defined already in the diagram file) with a repository association .The *sourceSymbol* and *destinationSymbol* have to be symbols already defined in the current map file. The *head* and *tail* are optional. *ConnectorLabel* too is optional and appears alongside the connector in the diagram.

Object Connector specification

Figure 12 shows a template for defining an object connector in accordance with this invention:
The *object connector* specification is for associating a connector symbol (specified in diagram file) with an object connector (specified as a meta object in the repository). The first three specifications are the same as

those in the *association connector* specifications. *object* specification is to map the connector to the repository object for the connector. *objectprop* specification has the same meaning as the *objectprop* specification in the *symbol* specification. The *sourceAssociation* and *destAssociation* specifications are to capture the names of the meta associations between the connector object and the source/destination (Since the connector is an object in the repository, it could have different meta-associations with the source and the destination).The *head* and *tail* specifications are used as a filter to select head / tail connector-ends. The filtering is based on a logical expression involving property values; when the expression evaluates to TRUE, the corresponding connector-end is used. It is possible to specify a default connector-end when no expression is satisfied. The *connectorLabel* specification is optional (like in the *association connector* specification) the difference is the label could also be a property of the connector object.

Figure 13 shows a template for defining a junction connector in accordance with this invention.

The oneToManyJunction shown in Figure 13 is for defining junction boxes .The *icon* specification is for specifying a diagram symbol to represent the junction box in the diagram .The *connectorIcon* specification is optional and is taken as a default for the various connectors required to connect the symbols to the junction box. For this implementation of the junction box, when one object (referred to as the parent) is connected to various objects (referred to as child), a junction box can be used such that, there is one connector between the junction box and the parent and different connectors between the junction box and each of the children. The *parent* specification identifies the parent symbol (has to be defined previously in the map file) and the diagram connector to be used to connect the parent symbol to the junction box - this will overwrite the default if there is one specified. This is an optional specification and if not specified, the default specification given in the beginning will be taken.

The *head* specification is used as a filter to select the connector-end for the connector from the junction-box to the parent. The filtering is based on a logical expression involving property values; when the expression evaluates to TRUE, the corresponding connector-end is used. It is possible to specify a default connector-end when no expression is satisfied. The *child* specification (there can be more than one of this kind) identifies the child *symbol* (has to be defined previously in the map file) and the connector to be used to connect the child to the junction box. The *child* specification also specifies whether the connection between the child and the parent in the repository is a meta association or there is another object through which the parent and child connect to each other. Accordingly, the relevant information has to be entered as in the case of object connector. The *connectorIcon* specification is used to specify the diagram connector to be used for the connector between the child symbol and the junction box - this will overwrite the default if there is one specified. This is an optional specification and if not specified, the default specification given in the beginning will be taken.

The *tail* specification is used as a filter to select the connector-end for the connector from the junction-box to the child. The filtering is based on a logical expression involving property values; when the expression evaluates to TRUE, the corresponding connector-end is used. It is possible to specify a default connector-end when no expression is satisfied.

Figure 14 shows a template for defining Nary connector in accordance with this invention.

The Nary connector shown in Figure 14 can be used to represent associations in case where a particular meta object is linked with a number of other meta objects through independent associations. Nary connector can have many parents and many children. All parents are connected to all children with independent associations.

In Use, the method and apparatus of this invention can create a generic diagram drawing tool that interprets the specifications shown in the drawings.

1. The drawing tool supports a drawing sheet window and an icon palette. The icon palette is populated with icons for symbols and connectors as defined in the diagram specification file.
2. When a symbol icon is picked from the icon palette and placed on the drawing sheet, a user can draw the symbol as specified in the diagram specification file, and the method allows it to be mapped to a model element which is an instance of the meta-object as specified in the map file. It is now made possible to select an existing model element or create a new model element. Model element selections respect the filtering constraints specified in the map file. The symbol's fields (text as well as list fields) are populated as specified in the map file.

3. When a connector icon representing an association connector is picked from the icon palette and placed between two symbols, the following happens:

- If the connector is not valid between the symbols, an error message is given.
- If the specified association already exists in the repository, nothing needs to be done, otherwise the association is added in the repository.

4. When a connector icon representing an object connector is picked from the icon palette and placed between two symbols, the following happens:

- If the connector is not valid between the symbols, an error message is given.
- If there already exists an object with associations as specified in the map file between the two objects represented by the symbols, then nothing needs to be done, otherwise a new object and the required associations need to be created in the repository.

5. When a connector icon representing a junction connector is picked up and placed between a junction box and a parent symbol, the following things happen:

- If the junction box has no connected child symbols, then nothing needs to be done. If the junction box has connected child symbols, then associations need to be created between the objects represented by the parent symbol and each of the child symbols if these associations do not already exist.

6. When a connector icon representing a junction connector is picked up and placed between a junction box and a child symbol, the following things happen:

- If the junction box has no connected parent symbol, then nothing needs to be done. If the junction box has a connected parent symbol, then an association needs to be created between the object represented by the parent symbol and the object represented by the child symbol if the association does not already exist.

7. When a connector icon representing an n-ary connector is placed between a parent and a child symbol, specified association or an object with appropriate associations is created between the objects representing the parent and the child. When an n-ary connector is placed between a parent symbol and an n-ary edge, specified associations or objects with appropriate associations are created between the object representing the parent and the objects represented by each of the child symbols.
8. For all connector types (association, object, junction and n-ary), annotations are filled as specified in the map file and connector ends are attached as specified.
9. When a containee symbol is placed inside a container symbol, an association, as specified in the map file is created between the objects representing the container and the containee symbols.

The above points specify how the diagrammer tool can interpret the diagram and map specification files. In addition, it provides common behavior like, grouping of symbols, copy/paste, move, alignment, etc.

Advantages of the invention

1. Programmable diagramming feature greatly enhances the productivity of repository implementation teams by enabling them to quickly implement diagrammatic notations for ever-evolving modeling abstractions.
2. It enables end-users to quickly implement visual modeling notations of their choice for enhanced meta-models, thus allowing them to fully

utilize and exploit the built-in extensibility features of model repository systems.

3. The proposed diagram model reflects the structure of the meta-meta-model, thus providing a close match between modeling abstractions and diagramming notations.

Although the invention has been described in terms of particular embodiments and applications, one of ordinary skill in the art, in light of this teaching, can generate additional embodiments and modifications without departing from the spirit of or exceeding the scope of the invention. Accordingly, it is to be understood that the drawings and descriptions herein are proffered by way of example to facilitate comprehension of the invention and should not be construed to limit the scope thereof.

Dated this 24th day of July 2001

[illegible]

Mohan Dewan

Of R K Dewan & Co

Applicants' Patent Attorney

TATA CONSULTANCY SERVICES
NAME :(TATA SONS LIMITED) NO. OF SHEETS : 10
NO. : / MUM / 01 SHEET NO. : 1

PROVISIONAL SPECIFICATION

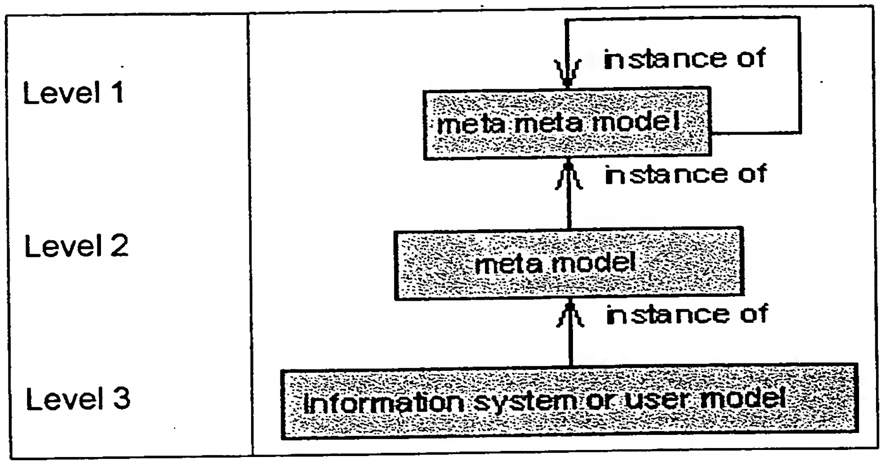

Figure 1- Modeling Framework

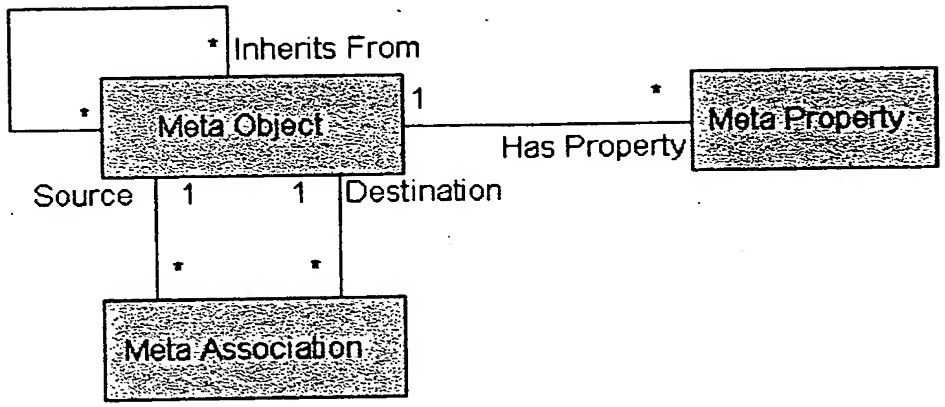

Figure 2- Meta meta model

[illegible]
DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES
NAME : (TATA SONS LIMITED)
NO. : / MUM / 01

NO. OF SHEETS : 10
SHEET NO. : 2

PROVISIONAL SPECIFICATION

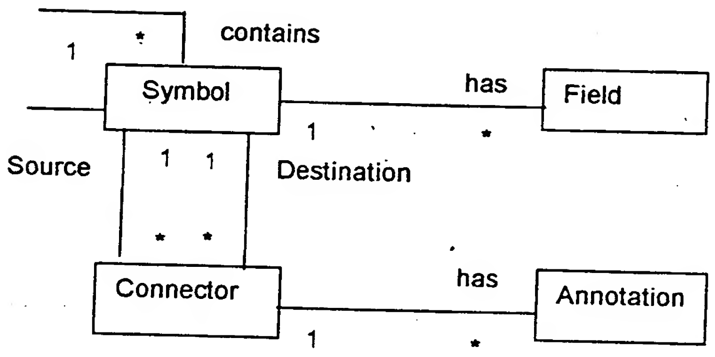

Figure 3 - Diagram Model

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES

NAME :(TATA SONS LIMITED)	NO. OF SHEETS : 10
NO. : / MUM / 01	SHEET NO. : 3

PROVISIONAL SPECIFICATION

Objects in Meta Meta Model	Properties of Object
MetaObject	Name, Description, AbstractConcerte

MetaProperty	DataType = Char, Number, Binary
	Size = Size of Char String and Number

MetaAssociation	Forward and Reverse Name Source and Destination Optionality = Association is optional or mandatory for source or destination object Source and Destination Cardinality = One or Many Owner of the Association = Source or destination object is the owner of the association

Associations in Meta Meta Model	Properties of Association
MetaObject Inherits From MetaObject	A MetaObject inherits associations and properties from another MetaObject. Many to Many; Optional
MetaObject Has MetaProperty	MetaObject has MetaProperties. One to Many; Optional
MetaObject SourceOf MetaAssociation	MetaObject is source of a MetaAssociation. One to Many; Mandatory for MetaAssociation
MetaObject DestinationOf MetaAssociation	MetaObject is destination of a MetaAssociation. One to Many; Mandatory for MetaAssociation

Figure - 4

Dewan

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

PROVISIONAL SPECIFICATION

```
SYMBOL Name, IconId, FrameX, FrameY {
        Property Section
        Shape Section
        Boundary Section
}
```

```
LINE StartX, StartY, EndX, EndY [: Options];
RECTANGLE TopLeftX, TopLeftY, Width, Height [: Options];
RRECTANGLE TopLeftX, TopLeftY, Width, Height, RWidth, RHeight [: Options]; --
rounded rectangle
ELLIPSE TopLeftX, TopLeftY, Width, Height [: Options];
ARC TopLeftX, TopLeftY, Width, Height, StartAngle, EndAngle [: Options];
CHORD TopLeftX, TopLeftY, Width, Height, StartAngle, EndAngle [: Options];
PIE TopLeftX, TopLeftY, Width, Height, StartAngle, EndAngle [: Options];
POLYGON x1, y1, x2, y2, x3, y3 [, xn, yn] [: Options];
BITMAP "BmpName", TopLeftX, TopLeftY, Width, Height [: Options]
TEXT TextId, TopLeftX, TopLeftY, Width, Height, InitText, MaxChar [: Options]
        TEXT fields are used for displaying properties of objects.
LIST ListId, TopLeftX, TopLeftY, Width, Height [: Options]
        LIST fields are used for displaying multi-column lists
```

Figure 5 - Symbol Definition

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

PROVISIONAL SPECIFICATION

```
ANNOTATION Name, IconId, FrameX, FrameY {
        Property Section
        Shape Section
}
```

Figure - 6 Annotation Definition

```
CONNECTOREND Name, IconId, FrameX, FrameY, AttachX, AttachY {
        Drawing commands
}
```

Figure - 7 ConnectorEnd Definition

```
CONNECTOR Name, IconId {
        Property Section
}
```

Figure - 8 Connector Definition

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES
NAME :(TATA SONS LIMITED)
NO. : / MUM / 01

NO. OF SHEETS : 10
SHEET NO. : 6

PROVISIONAL SPECIFICATION

```
symbol <symbol> {
        icon = <IconId>
        object = <repos meta object name>
        objectprop {
                <repos meta prop name> = <value> ....
        }
        symbolprop {
                <fieldId> = prop(<repos meta prop name>) |
                        assoc(<repos object name>,<repos meta assoc name>) |
                        script ("OSC file name") ...
        [listControl "<fieldId>"
        {
                [assocExpr = <repos meta assoc name>.<repos meta object
name>.[<repos meta
 assoc name>.<repos meta object name>...]
                colSpecs =
                {
                        "<column header>" = <repos meta prop name> | <repos
                meta assoc name>.<repos meta object name>.[<repos meta assoc
                name>.<repos meta object name>...].<repos meta prop name> |
                        script ("script file name")
                                ...
                }
        }]
        }
}
```

Figure - 9
Template for mapping a symbol icon to an object in the model

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

PROVISIONAL SPECIFICATION

```
object connector <connector>
{
          connectorIcon = <Connector IconId>
          sourceSymbol = <symbol>       [<symbol>...]
        destinationSymbol = <symbol>[<symbol>...]
        object = <repos meta object name>
        objectprop
        {
                  <repos meta prop name> = <value> ....
        }
        sourceAssociation =  <repos meta assoc name>
        destAssociation = <repos meta assoc name>
        head
        {
        <repos meta prop value expression> => <ConnectorEnd IconId> ....
                                                [ConnectorEnd IconId]
          }
          tail
        {
        <repos meta prop value expression> => < ConnectorEnd IconId > .....
                  [ConnectorEnd IconId]
        }
        [connectorLabel = "<value>" | <repos meta prop name> |  script ("script file
name") ]
        [connectorLabel {
                  "<Annotation IconId>" = "<value>" |
                  "<Annotation IconId>" = <repos meta prop name> |
                  "<Annotation IconId>" = script ("script file name")
                  ....

        }]
}
```

Figure - 12 Template for defining an object connector

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES
NAME :(TATA SONS LIMITED)
NO. : / MUM / 01

NO. OF SHEETS : 10
SHEET NO. : 7

PROVISIONAL SPECIFICATION

```
containerSpec <symbol> {
        container {
                symbol = <symbol> [<symbol> ...]
        }
        containee {
                symbol = <symbol> [<symbol> ...]
                association = <repos meta assoc name>
        }
        [containee {
                symbol = <symbol> [<symbol> ...]
                association = <repos meta assoc name>
        } ...
        ]
}
```

Figure - 10 Template for mapping a container with containees

```
association connector <connector> {
        connectorIcon = <Connector IconId>
        sourceSymbol = <symbol> [<symbol>...]
        destinationSymbol = <symbol> [<symbol>...]
        association = <repos meta assoc name>
        [
          head = <ConnectorEnd IconId>
          tail = < ConnectorEnd IconId >
        ]
        [connectorLabel = "<value>" |
                script ("script file name")
        ]
        [connectorLabel {
                "<Annotation IconId>" = "<value>" |
                "<Annotation IconId>" = script("script file name")
                ...
        }]
}
```

Figure - 11
Template for defining an association connector

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES
NAME :(TATA SONS LIMITED)
NO. : /MUM/01

NO. OF SHEETS : 10
SHEET NO. : 9

PROVISIONAL SPECIFICATION

```
oneToManyJunction <symbol>
{
        icon = <IconId>
        [connectorIcon = <Connector IconId>]
        parent
        {
                symbol = <symbol>[<symbol>...]
                [connectorIcon = < Connector IconId >]
                head
                {
                        <repos meta prop value expression>=> <ConnectorEnd
IconId> ....
                                                            [ConnectorEnd IconId]
                }
        }
        child {
                symbol = <symbol>[<symbol>...]
                association = <repos meta assoc. name> |
                objectConnector {
                        object = <repos meta object name>
                        objectprop {
                                <repos meta prop name> = <value> ....
                        }
                        parentAssociation = <repos meta assoc name>
                        childAssociation = <repos meta assoc name>
                }
                [connectorIcon = < Connector IconId >]
                tail {
                        <repos meta prop value expression>=> < ConnectorEnd
IconId > ....
                        [ConnectorEnd IconId]
                }
        } ....
}
```

Figure - 13 Template for defining a junction connector

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY

TATA CONSULTANCY SERVICES
NAME :(TATA SONS LIMITED)   NO. OF SHEETS : 10
NO. :   / MUM / 01   SHEET NO. : 10

PROVISIONAL SPECIFICATION

```
naryConnector <connectorName> {
        icon = "<Connector IconId>"
        parent {
                symbol = <symbol> [<symbol> ...]
                [head = "<ConnectorEnd IconId>"]
                [head { "<ConnectorEnd IconId>" }]
        }
        child
        {
                symbol = <symbol> [<symbol> ...]
                association = <repos meta assoc name> |
                objectConnector {
                        object = <repos meta object name>
                        [objectprop {
                                <repos meta prop name> = "<value>" ...
                        }]
                        parentAssociation = <repos meta assoc name>
                        childAssociation = <repos meta assoc name>
                }
                [tail = "<ConnectorEnd IconId>"]
                [tail {
                        <repos meta prop value expression> => "<ConnectorEnd
IconId>" ... |
                        "<ConnectorEnd IconId>"
                }
                ]
        }
```

Figure - 14   Template for defining Nary connector

[illegible]

DR. MOHAN DEWAN
APPLICANT'S PATENT ATTORNEY